IJournals: International Journal of Software & Hardware Research in Engineering
ISSN-2347-4890
Volume 4 Issue 1 January, 2016

Load Balancing with Minimum Task

Ranjan Kumar Mondal
Dept. of CSE,
University of Kalyani,
WB, India
ranjan@klyuniv.ac.in

Payel Ray
Dept. of CSE,
University of Kalyani, WB,
India
payelray009@gmail.com,

Enakshmi Nandi
Dept. of CSE,
University of Kalyani, WB,
India
pamelaroychowdhurikalyani@gmail.com,

Debabrata Sarddar
Assistant Professor,
Dept. of CSE,
University of Kalyani,
dsarddar1@gmail.com

Abstract.

There are various types of heterogeneous nodes in a cloud system. Namely, each node havingdifferent capability executes task of a system; hence, only consider the CPU remaining of the node is not enough when a node is chosen to execute a task. Therefore, how to select an efficient node to execute a task is very important in a cloud computing.

In this paper, we propose a new scheduling algorithm that choose a suitable node with its average task. It is very easy way to select an appropriate node. This approach can provide efficient utilization of computing resources and maintain the load balancing in cloud computing environment.

Keywords:Cloud Computing, Load Balancing, Distributed System, Threshold.

1.INTRODUCTION

Load balancing in cloud computing is a new challenge nowadays. Always a distributed solution is required because it is not always practically feasible or cost efficient to maintain one or more idle services just as to fulfill the required demands. Jobs can't be assigned to appropriate servers and clients individually for efficient load balancing as cloud is a very complex structure and components are present throughout a wide spread area.

Load balancing algorithms are classified as static load balancing algorithms and dynamic load balancing algorithms. Static load balancing algorithms are suitable for homogeneous and stable environments and can produce very good results in these environments. However, they are usually not flexible and cannot match the dynamic changes to the attributes during the execution time. Dynamic load balancing algorithms are more flexible and take into consideration different types of attributes in the system both prior to and during run-time[1]. Load balancing is the process of improving the performance of system through a redistribution of load among processor.

2. LITERATURE REVIEW

A set of heuristic algorithms has been proposed to schedule meta-tasks to heterogeneous computing systems. It is assumed that the heuristic derive mapping statically for the collection of independent meta-task.The scheduling problem is computationally hard even though there are no data dependencies among the jobs.

2.1. Opportunistic Load Balancing

OLB assigns each task, in arbitrary order, to the next machine that is expected to be available, regardless of the task's expected execution time on that machine. The intuition behind OLB is to keep all machines as busy as possible [2].

2.2. Minimum Execution Time

In contrast to OLB, MET assigns each task, in arbitrary order, to the machine with the best expected execution time for that task, regardless of that machine's availability. The motivation behind MET is to give each task to its best machine. This can cause a severe load imbalance across machines [3].

2.3. Minimum Completion Time

MCT assigns each task, in arbitrary order, to the machine with the minimum expected completion time for that task. This causes some tasks to be assigned to machines that do not have the minimum execution time for them [1].

2.4. Min-min

Min-min heuristic uses minimum completion time (MCT) as a metric, meaning that the task which can be done the earliest is given priority. This heuristic begins with the set U of all unmapped tasks. Then the set of minimum completion times (M), is found.

M consists of one entry for each unmapped task. Next, the task with the overall minimum completion time from M is selected and assigned to the corresponding machine and the workload of the selected machine will be updated. And finally the newly mapped task is removed from U and the process repeats until all tasks are mapped [4].

2.5. Max-Min

The Max-min heuristic is very similar to Min-min and its metric is MCT too. It begins with the set U of all unmapped tasks. Then, the set of minimum completion times (M) is found as mentioned in previous section. Next, the task with the overall maximum completion time from M is selected and assigned to the corresponding machine and the workload of the selected machine will be updated. And finally the newly mapped task is removed from U and the process repeats until all tasks are mapped [5].

3. PROBLEM DEFINITION

A heterogeneous computing environment is composed of computing resources where these resources can be a single PC, a cluster of workstations or a supercomputer. The main goal of the task scheduling in HC environments is the efficiently allocating tasks to machines. The efficiency means that we are interested to allocate tasks as fast as possible and optimizing the makespan and execution time objectives. Tasks are originated from different users/applications, are independent. We use the ETC (Expected Time to Compute) matrix model introduced by Shoukat et al. for formulating the problem [20]. It is assumed that an accurate estimate of the expected execution time for each task on each machine is known prior to execution and contained within an ETC matrix. Each row of the ETC matrix contains the estimated execution times for a given task on each machine. Similarly, each column of the ETC matrix consists of the estimated execution times of a given machine for each task. ETC [i,j] is the expected execution time of task i in machine j. For the simulation studies, characteristics of the ETC matrices were varied in an attempt to represent a range of possible heterogeneous environments. Using the ETC matrix model, the scheduling problem can be defined as follows:

1. A number of independent tasks to be allocated to the available resources. Because of Non-preemptive scheduling, each task has to be processed completely in a single machine.
2. Number of machines is available to participate in the allocation of tasks.
3. The workload of each task
4. The Computing capacity of each resources
5. Ready[m] represents the ready time of the machine after completing the previously assigned tasks.
6. ETC matrix of size m×n where n represents the number of tasks and m represents the number of machines.

4. THE PROPOSED METHOD

There are several heterogeneous nodes in a cloud computing system. All nodes have no capability to execute same task; hence, only consider the CPU remaining of the node is not enough when a node is chosen to execute a task. Therefore, how to select an efficient node to execute a task is very important in a cloud computing.

Due to the task maybe has different characteristic for user to pay execution. Hence it is need some of the resources of specific, for instance, when implementorganism sequence assembly, it is probable have to big requirement toward memory remaining. And in order to reach the best efficient in the execution each tasks, so we will aimed tasks property to adopt a different condition decision variable in which it is according to resource of task requirement to set decision variable.

4.1. Method

Step 1: Remove higher subtasks of all nodes from the system until every node has each subtask only.

Step 2: Select that minimum subtask to corresponding node.

Step 3: Dispatch that nodes that are already assigned by subtask from the system.

Step 4: If there are more than same subtask then choose other node that task has minimum value.

Step 5: When all subtask are assigned to corresponding nodes then stop the procedure.

Step 6: Repeat Step 2 to Step 4, until all tasks have been completed totally.

In the following section, an example to be executed by using the proposed algorithm is given.

Case Study I:

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	18	14	38	26
A_2	14	12	24	18
A_3	26	18	66	42
A_4	19	20	24	36

Step 1: Remove higher subtasks until ever node has each subtask only.

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	18	14	~~38~~	26
A_2	14	12	24	18
A_3	~~26~~	18	~~66~~	~~42~~
A_4	19	20	24	~~36~~

Step 1: Node N14 has only subtask i.e. 18 it is assigned to that node.

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	18	14	~~38~~	~~26~~
A_2	14	12	24	18
A_3	~~26~~	18	~~66~~	~~42~~
A_4	19	20	24	~~36~~

Similarly N12 is assigned by A3.

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	**18**	14	~~38~~	~~26~~
A_2	14	12	24	**18**
A_3	~~26~~	**18**	~~66~~	~~42~~
A_4	~~19~~	20	24	~~36~~

Here is a conflict that N13 has to same subtask that has 24. Now we have to choose the subtask that has minimum value from N11 that is 18.

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	**18**	14	~~38~~	~~26~~
A_2	14	12	24	**18**
A_3	~~26~~	**18**	~~66~~	~~42~~
A_4	~~19~~	20	24	~~36~~

Finally we select last A4 task of N13.

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	**18**	14	~~38~~	~~26~~
A_2	14	12	24	**18**
A_3	~~26~~	**18**	~~66~~	~~42~~
A_4	~~19~~	20	**24**	~~36~~

Final Result following

Node \ task	N_{11}	N_{12}	N_{13}	N_{14}
A_1	**18**	14	38	26
A_2	14	12	24	**18**
A_3	26	**18**	66	42
A_4	19	20	**24**	36

5. Comparison:

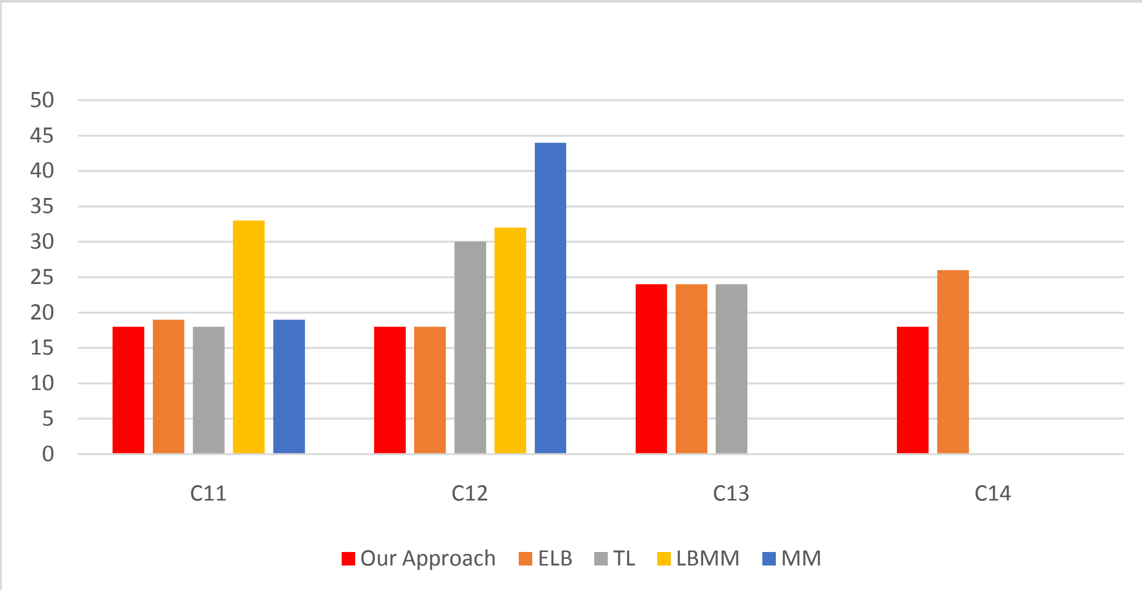

Fig 1.The comparison of completion time of each task at different node for case study I.

6.CONCLUSION

In this paper, we proposed an efficient scheduling algorithm, LBMT, for the cloud computing network to assign tasks to computing nodes according to their resource capability. Similarly, our approach can achieve better load balancing and performance than other algorithms, such as LB3M, MM and LBMM from the case study.

In this paper, we have presented a new scheduling algorithm for scheduling. The goal of the scheduler in this paper is minimizing makespan and maximizes resources utilization.

7. ACKNOWLEDGMENT:

We would like to express our gratitude to Dr. Kalyani Mali, Head of Department, Computer Science and Engineering of University of Kalyani. Without her assistance and guidance, we would not have been able to make use of the university's infrastructure and laboratory facilities for conducting our research.

REFERENCES

[1]. GautamShroff, "Enterprise Cloud Computing," Cambridge University Press, June, 2011.

[2]. Wang, S. C., Yan, K. Q., Liao, W. P., & Wang, S. S. (2010, July). Towards a load balancing in a three-level cloud computing network. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on* (Vol. 1, pp. 108-113). IEEE.

[3]. Shan, Ming-Chien, and Marguerite C. Murphy. "Method of automatically controlling the allocation of resources of a parallel processor computer system by calculating a minimum execution time of a task and scheduling subtasks against resources to execute the task in the minimum time." U.S. Patent No. 5,325,525. 28 Jun. 1994.

[4]. Braun, T. D., Siegel, H. J., Beck, N., Bölöni, L. L., Maheswaran, M., Reuther, A. I.,.& Freund, R. F. (2001). A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. Journal of Parallel and Distributed computing, 61(6), 810-837. L. Garces-Erice, et al., "Hierarchical Peer-to-peer Systems," Proceedings of ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par), 2003.

[5]. Nace, D., &Pióro, M. (2008). Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial. *Communications Surveys & Tutorials, IEEE, 10*(4), 5-17.

[6]. Braun, T. D., Siegel, H. J., Beck, N., Bölöni, L. L., Maheswaran, M., Reuther, A. I., Robertson,

J. P., Theys, M. D., Yao, B., Hensgen, D., Freund, R. F.: A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, vol. 61, pp. 810—837, (2001)

[7]. Wang, S. C., Yan, K. Q., Liao, W. P., Wang, S. S.: Towards a Load Balancing in a threelevel cloud computing network. In: Computer Science and Information Technology, pp. 108—113, (2010).

[8]. Ranjan KumarMondal, Enakshmi Nandi, and Debabrata Sarddar. "Load Balancing Scheduling with Shortest Load First." *International Journal of Grid and Distributed Computing* 8.4 (2015): 171-178.

[9]. Ranjan Kumar Mondal, Debabrata Sarddar "Load Balancing with Task Subtraction of Same Nodes". International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online) Vol. 3, Issue 4, pp: (162-166), Month: October - December 2015.

[10]. Ranjan Kumar Mondal, Payel Ray, Debabrata Sarddar "Load Balancing with Task Division and Addition". International Journal of Scientific Research Engineering & Technology (IJSRET), ISSN 2278 – 0882, Volume 5, Issue 1, January 2016: 15-19.

AUTHORS PROFILE

Ranjan Kumar Mondal received his M.Tech in Computer Science and Engineering from University of Kalyani, Kalyani, Nadia; and B.Tech in Computer Science and Engineering from Government College of Engineering and Textile Technology, Berhampore, Murshidabad, West Bengal under West Bengal University of Technology, West Bengal, India. At present, he is a Ph.D research scholar in Computer Science and Engineering from University of Kalyani. His research interests include Cloud Computing, Wireless and Mobile Communication Systems.

Payel Ray received her M.Tech in Computer Science and Engineering from Jadavpur University, Jadavpur, India; and B.Tech in Computer Science & Engineering from Murshidabad College of Engineering and Technology, Berhampore, Murshidabad, West Bengal under West Bengal University of Technology, West Bengal, India. At present, she is a Ph.D research scholar in Computer Science and Engineering from University of Kalyani. Her research interests include Cloud Computing, Wireless Adhoc and Sensor Network and Mobile Communication Systems.

Enakshmi Nandi received her M. Tech in VLSI and Micro-electronics from Techno India, Salt Lake, West Bengal and B. Tech in Electronics and Communication Engineering from JIS College of Engineering, West Bengal under West Bengal University of Technology, West Bengal, India. At present, she is Research scholar in Computer Science and Engineering from University of Kalyani. Her research interests include Cloud Computing, Mobile Communication system, Device and Nanotechnology.

Debabrata Sarddar is an Assistant Professor at the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India. He completed his PhD from Jadavpur University. He did his M. Tech in Computer Science & Engineering from DAVV, Indore in 2006, and his B.E in Computer Science & Engineering from NIT, Durgapur in 2001. He has published more than 75 research papers in different journals and conferences. His research interests include Wireless and Mobile Systems and WSN, and Cloud computing.